**Практика «Нестатические методы»**

Вы вдруг поняли, что не очень-то удобно писать имя класса Geometry при выполнении любой операции с векторами и сегментами. Однако, отказаться от этого класса вы не можете, потому что за те несколько минут, пока вы сдавали предыдущую задачу, вашу библиотеку скачали и начали использовать в своих проектах тысячи человек.

Поэтому вы решили сохранить этот класс, но добавить методы Vector.GetLength(), Segment.GetLength(), Vector.Add(Vector), Vector.Belongs(Segment) и Segment.Contains(Vector) не вместо, а вместе с соответствующими методами класса Geometry.

Сделайте это! Каждый из этих методов должен вызывать уже существующий метод класса Geometry, чтобы не дублировать код.

Вся функциональность предыдущего этапа должна остаться!

// Вставьте сюда финальное содержимое файла VectorTask.cs

**Программа:**

**Содержимое файла Program.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using GeometryTasks;

namespace GeometryTasks

{

class Program

{

public static Vector

v1 = new Vector { X = 3.2, Y = 8.5 },

v2 = new Vector { X = 5.1, Y = 3.4 },

VSum = new Vector { X = 0, Y = 0 };

public static Segment

sgm = new Segment

{

Begin = new Vector() { X = 8.1, Y = 11.36 },

End = new Vector() { X = 15.74, Y = 19.42 }

};

public static void Main(string[] args)

{

Console.WriteLine("Длина вектора v1={0}", Geometry.GetLength(v1));

Geometry.Add(v1, v2);

Console.WriteLine("Суммирующий вектор х={0} y={1}", Geometry.VSum.X, Geometry.VSum.Y);

Console.WriteLine();

Console.WriteLine("Длина сегмента sgm={0}", Geometry.GetLength(sgm));

Console.WriteLine(Geometry.IsVectorInSegment(v1, sgm));

Console.WriteLine();

Console.WriteLine(v1.Belongs(sgm));

Console.WriteLine();

Console.WriteLine(sgm.Contains(v1));

Console.ReadKey();

}

}

}

**Содержимое файла VectorTask.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using GeometryTasks;

namespace GeometryTasks

{

public class Vector

{

public double X;

public double Y;

public double GetLength()

{

return Geometry.GetLength(this);

}

public Vector Add(Vector v)

{

return Geometry.Add(this, v);

}

public bool Belongs(Segment s)

{

return Geometry.IsVectorInSegment(this, s);

}

}

public class Segment

{

public Vector Begin;

public Vector End;

public double GetLength()

{

return Geometry.GetLength(this);

}

public bool Contains(Vector v)

{

return Geometry.IsVectorInSegment(v, this);

}

}

public static class Geometry

{

public static Vector

VSum = new Vector { X = 0, Y = 0 };

public static Segment

Sg = new Segment

{

Begin = new Vector { X = 0, Y = 0 },

End = new Vector { X = 1, Y = 1 }

};

public static double DlinaV, S, S1;

public static bool B = false;

public static double GetLength(Vector v1)

{

S = v1.X \* v1.X + v1.Y \* v1.Y;

if (S > 0)

{

return DlinaV = Math.Sqrt(S);

}

else return 0;

}

public static Vector Add(Vector v1, Vector v2)

{

VSum.X = v1.X + v2.X;

VSum.Y = v1.Y + v2.Y;

return VSum;

}

public static double GetLength(Segment sgm)

{

S1 = (sgm.End.X - sgm.Begin.X) \* (sgm.End.X - sgm.Begin.X) +

(sgm.End.Y - sgm.Begin.Y) \* (sgm.End.Y - sgm.Begin.Y);

return Math.Sqrt(S1);

}

public static bool IsVectorInSegment(Vector v, Segment sg)

{

B = ((v.X - sg.Begin.X) \* (v.X - sg.End.X) <= 0) && ((v.Y - sg.Begin.Y) \* (v.Y - sg.End.Y) < 0);

if (((v.X == sg.Begin.X) || (v.X == sg.End.X)) && ((v.Y == sg.End.Y) || (v.Y == sg.Begin.Y)))

return true;

else

return B;

}

}

}